

15 FEBRUARY 2000

Europäisches
PatentamtEuropean
Patent OfficeOffice européen
des brevets

GB 00 / 504

EJU

REC'D 28 FEB 2000

WIPO

PCT

Bescheinigung

Certificate

Attestation

Die angehefteten Unterla-
gen stimmen mit der
ursprünglich eingereichten
Fassung der auf dem näch-
sten Blatt bezeichneten
europäischen Patentanmel-
dung überein.

The attached documents
are exact copies of the
European patent application
described on the following
page, as originally filed.

Les documents fixés à
cette attestation sont
conformes à la version
initialement déposée de
la demande de brevet
européen spécifiée à la
page suivante.

Patentanmeldung Nr. Patent application No. Demande de brevet n°

99306416.1

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH RULE 17.1(a) OR (b)

Best Available Copy

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

I.L.C. HATTEN-HECKMAN

DEN HAAG, DEN
THE HAGUE, 13/10/99
LA HAYE, LE

400 000

0000 000000

THIS PAGE BLANK (USPTO)



Europäisches
Patentamt

European
Patent Office

Office européen
des brevets

**Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation**

Anmeldung Nr.:
Application no.: 99306416.1
Demande n°:

Anmeldetag:
Date of filing: 13/08/99
Date de dépôt:

Anmelder:
Applicant(s):
Demandeur(s):
Hewlett-Packard Company
Palo Alto, California 94304
UNITED STATES OF AMERICA

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:

Communications between modules of a computing apparatus

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat:
State:
Pays:

Tag:
Date:
Date:

Aktenzeichen:
File no.
Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:

/

Am Anmeldetag benannte Vertragsstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE
Etats contractants désignés lors du dépôt:

Bemerkungen:
Remarks:
Remarques:

THIS PAGE BLANK (USPTO)

- 1 -

TITLE

Communications between Modules of a Computing Apparatus

DESCRIPTION

This invention relates to computing apparatuses and is concerned in particular with improving trust and security between various modules of a computing apparatus.

Concerns over the integrity of computer platforms arise because commercial client platforms are often in environments which are vulnerable to unauthorised modification, either
5 by software loaded by the user or by software loaded *via* a network connection. Therefore users may be reluctant to trust a platform for manipulation of critical data, and it is desirable to improve the level of security within the computer platform.

Weakness arises from the fact that conventional functional modules (CPU, memory, hard disk drive, keyboard, CDROM, DVD, smartcard reader, ISA card, EISA card, PCI card,
10 etc.) of a computer platform do not validate information communicated between those modules, and communications with a module are not confidential. The main reasons for this weakness are that:

- the main 'computing engine' of a computer platform is physically distributed within the platform (separate CPU, memory, long term storage, etc.), and many individual
15 distributed components of that main computing engine do not have the functionality to secure the data they communicate;
- communications between both the elements of that distributed engine and with other functional modules are done using shared communication paths (a shared communication infrastructure); and
- 20 • the services of functional modules are potentially shared by several other functional modules.

Hence rogue software on the main distributed computing engine, or on another functional module, can eavesdrop on data, and perform or cause inappropriate manipulation of data.

25 In any shared environment, secure communications (having authentication, integrity, confidentiality, etc.) are possible if functional modules have their own self-contained computing engines that are capable of cryptographic security functions. Such engines would allow functional modules to authenticate the source of data, verify the integrity of data, and provide

- 2 -

confidentiality of data. Many functional modules in a computer platform already have self-contained computing engines, and it would be possible to modify those engines (physically and/or by programming) to provide the necessary cryptographic security functions. Such modifications, *per se*, will be obvious to those skilled in the art of security and electronics.

- 5 Even so, a remaining problem is that all such cryptographic processes rely upon the use of secrets, commonly called keys. Keys must be distributed. Each functional module must have its own secrets and access to the appropriate secrets of other functional modules. These properties and difficulties, and the nature and use of keys, are, *per se*, well known to those skilled in the art of information security.

- 10 A prior patent application (EP 99 301100.0) described the use of a trusted component or module in a computer platform to enable verification of the integrity of a computer platform by the reliable measurement and reliable reporting of integrity metrics. A trusted module is essentially immune to unauthorised modification or inspection of internal data. It is physical to prevent forgery, tamper-resistant to prevent counterfeiting, and has crypto functionality to
- 15 securely communicate at a distance. Methods of building trusted modules, *per se*, will be apparent to those skilled in the art. The trusted module uses cryptographic methods to give itself a cryptographic identity and to provide authenticity, integrity, confidentiality, guard against replay attacks, make digital signatures, and use digital certificates as required. These and other crypto methods and their initialisation are, *per se*, well known to those skilled in the
- 20 art of security.

- In accordance with the present invention, there is provided a computing apparatus, comprising a trusted hardware module; a plurality of further hardware modules; a shared communication infrastructure (such as normal EISA and PCI buses) by which the modules can communicate with each other; and a first communication path, distinct from the communication
- 25 infrastructure, by which a first one of the further modules can communicate directly with the trusted module but cannot communicate directly with any other of the further modules. (It should be noted that the first further module may or may not itself be a trusted module in its own right.) Accordingly, the trusted module and first further module can communicate, without the need for encryption and decryption, and without any possibility of any other part of the
- 30 apparatus eavesdropping on the communication. The first communication path may be implemented as a single dedicated physical wire that permits serial communications, or multiple wires, or optical connections, or very short distance wireless transmissions on different frequencies, and so on. The invention enables the private, implicitly authenticated, communication of information between the trusted module and the first further module. This
- 35 enables communication of private information, or information whose reliability is paramount.

This includes, for example, data *per se*, or a command, an indication, a flag, or the checksum of data that is to be communicated over the shared communication infrastructure.

Preferably, the trusted module and the first further module each include a respective computing engine which partakes in the direct communication *via* the first communication path.

5 (It should be understood that, in this context, the phrase "computing engine" means any device capable of manipulating data, including programmed devices and hardwired devices.) Not all functional blocks require the addition of a separate such computing engine. A hard disk drive, for example, may already incorporate a computing engine for the purposes of controlling the hard disk drive and simply require the addition of extra instructions to provide control of the

10 additional communication path. On the other hand, a normal PC's CPU (such as Intel's Pentium processor) is obviously a computing engine, but has no obvious separate computing engine. It may therefore be provided with an additional computing engine, or some other method to allow the CPU's general purpose computing engine to be used as the 'separate' computing engine. For example, an existing CPU can be modified to include a new instruction

15 and new hardware that communicates data over the protected communication path.

The first further module is preferably operable to supply to the trusted module a request for operation on data, and, in response to such a request, the trusted module is operable to generate a response and to supply the response to the first further module *via* the first communication path and not *via* the shared communication infrastructure. In this case, the

20 trusted module preferably includes means for storing policy information regarding such operations which can and/or cannot be permitted, and is operable to generate the response with reference to the policy information. This improves the level of trust in services provided by one module to another. In particular, note that the first further module can take advantage of the trusted module. The trusted module can, for example, act as a crypto-coprocessor or

25 'hardware' crypto API, but also contains other non-crypto functions whose trustworthiness is vital, such as computer programs described in other patent applications. The first further module can send data and a request to the trusted module using the shared communication infrastructure. The trusted module can act on the request using the supplied data and report the result back to the first further module over the first communication path. This allows a network

30 interface card, for example, to verify the origin and integrity of external messages without having to securely store (and hide) the necessary keys. Alternatively, the request may be to execute certain secret or protected functions (held within the trusted module) on the supplied data. In all cases, the fact that the response is received using the first communication path indicates to the first further module that the proper trusted module actually received the request,

- 4 -

and the response received on the first communication path is actually the judgement of the proper trusted module.

The trusted module is preferably operable to generate an encryption and/or decryption key and to supply that key to the first further module *via* the first communication path and not *via* the shared communication infrastructure. In this case, the first further module preferably is operable to use the key for encryption and/or decryption of data communicated *via* the shared communication infrastructure. Accordingly, the key is kept secret as between the trusted module and the first further module, without the possibility of any other part of the apparatus eavesdropping on the communication of the key. The computing engines in the trusted module and the first further module can exchange keys in plain text along the first communication path. If it is even unacceptable for keys to appear in clear on the first communication path, the trusted module and the first further module can use the Diffie-Hellman protocol to set-up keys therebetween. In the Diffie-Hellman case, the use of the first communication path simply provides implicit proof of identity of the 'other' entity engaging in the Diffie-Hellman process. The apparatus can also be used for maintenance and replacement of keys. Such key distribution methods, *per se*, are well known to those skilled in the art of information security. Once keys have been exchanged, those keys can be used to provide security for communications using the shared communication infrastructure. Such secure communication methods, *per se*, are well known to those skilled in the art of information security. Note that the generation/distribution of keys is useful even when all of the further modules are also trusted modules.

The trusted module is preferably operable to generate a challenge and to supply the challenge to the first further module *via* the first communication path or *via* the shared communication infrastructure using encryption set up using the first communication path. In this case, in response to the challenge, the first further module is preferably operable to generate a response and to supply the response to the trusted module *via* the first communication path or *via* the shared communication infrastructure using encryption set up using the first communication path; and the trusted module is operable to use the response in generating an integrity metric of the apparatus. This enables hardware 'watchdog' functionality, where the trusted module can at intervals challenge the first further module over the first communication path or using secure communications set-up using the first communication path. The first further module responds to the challenge with a reply using the first communication path (or secure communications over some other path set up using the first communication path). Such watchdog challenges may be done as a result of receiving an integrity challenge as described in the prior patent application mentioned above.

- 5 -

In one example of the first further module, particularly applicable when it is a network interface module, the first further module preferably has a zone for private data and a zone for non-private data; and the first further module is operable to supply and/or receive data from/for the private data zone *via* the first communication path and not *via* the shared communication infrastructure. In this case, the first further module is preferably operable to supply and/or receive data from/for the non-private data zone *via* the shared communication infrastructure. Also, the first further module preferably has an interface between the private and non-private data zones which is operable to inhibit the passing of unnecessary data from the private data zone to the non-private data zone. Accordingly, the apparatus can be neatly partitioned into safe and unsafe, or non-private and private, data zones.

The apparatus may, of course, include a second communication path, distinct from the communication infrastructure and the first communication path, by which a second one of the further modules (such as a main processor unit of the apparatus or a non-volatile data storage module) can communicate directly with the trusted module. In this case, the second further module preferably cannot communicate directly *via* the second communication path with any other of the further modules.

In this case, the first further module is preferably operable to supply to the trusted module a request for a transfer of data between the first and second further modules; and, in response to such a request, the trusted module is operable to generate a response and to supply the response to the first or second further module *via* the first or second communication path, as the case may be, and not *via* the shared communication infrastructure. In this case, the trusted module preferably includes means for storing policy information regarding such transfers which can and/or cannot be permitted, and is operable to generate the response with reference to the policy information. Also, in response to an appropriate such transfer response, the first or second further module is preferably operable to supply the data to the trusted module *via* the first or second communication path, as the case may be, and, in response to the receipt of such data, the trusted module is operable to relay the data to the second or first further module, as the case may be, *via* the second or first communication path, as the case may be.

The apparatus may, of course, include at least a third communication link, distinct from the communication infrastructure and the other communication links, by which at least a third one of the further modules can communicate directly with the trusted module. In this case, the third further module preferably cannot communicate directly *via* the third communication path with any other of the further modules.

Accordingly, the trusted module can have a switching capability, and the individual communication paths can be used as a 'star' communications network, enabling communications between the further modules under the control of the trusted module. The trusted module can route or block information between further modules according to some
5 policy stored in the trusted module. In particular, since the communication network is a star, the trusted module can present information to the concerned modules only, thus denying other modules the opportunity to eavesdrop.

A specific embodiment of the present invention will now be described, purely by way of example, with reference to the accompanying drawings, in which:

- 10 Figure 1 is a diagram which shows the motherboard of computing apparatus adapted to include a trusted device and as described in the prior patent application mentioned above;
- Figure 2 is a diagram which shows in more detail the trusted device shown in Figure 1;
- Figure 3 is a diagram which shows in the contents of a certificate stored in the trusted
15 device;
- Figure 4 is a diagram, which shows the features of a measurement function responsible for acquiring an integrity metric;
- Figure 5 is a flow diagram which illustrates the steps involved in acquiring an integrity metric of the computing apparatus;
- 20 Figure 6 is a flow diagram which illustrates the steps involved in establishing communications between a trusted computing platform and a remote platform including the trusted platform verifying its integrity;
- Figure 7 is a schematic block diagram of a host computer system embodying the invention;
- 25 Figure 8 is a schematic block diagram of a trusted module in the system of Figure 7;
- Figures 9 to 12 show parts of the system of Figure 7 to illustrate various communication methods employed therein; and
- Figure 13 illustrates the format of a protocol data unit used in the system of Figure 7.

Before describing the embodiment of the present invention, the computing platform incorporating a trusted device which is the subject of the prior patent application mentioned above will firstly be described with reference to Figures 1 to 6.

5 That application describes the incorporation into a computing platform of a physical trusted device or module whose function is to bind the identity of the platform to reliably measured data that provides an integrity metric of the platform. The identity and the integrity metric are compared with expected values provided by a trusted party (TP) that is prepared to vouch for the trustworthiness of the platform. If there is a match, the implication is that at least part of the platform is operating correctly, depending on the scope of the integrity metric.

10 A user verifies the correct operation of the platform before exchanging other data with the platform. A user does this by requesting the trusted device to provide its identity and an integrity metric. (Optionally the trusted device will refuse to provide evidence of identity if it itself was unable to verify correct operation of the platform.) The user receives the proof of identity and the integrity metric, and compares them against values which it believes to be true.
15 Those proper values are provided by the TP or another entity that is trusted by the user. If data reported by the trusted device is the same as that provided by the TP, the user trusts the platform. This is because the user trusts the entity. The entity trusts the platform because it has previously validated the identity and determined the proper integrity metric of the platform.

Once a user has established trusted operation of the platform, he exchanges other data
20 with the platform. For a local user, the exchange might be by interacting with some software application running on the platform. For a remote user, the exchange might involve a secure transaction. In either case, the data exchanged is 'signed' by the trusted device. The user can then have greater confidence that data is being exchanged with a platform whose behaviour can be trusted.

25 The trusted device uses cryptographic processes but does not necessarily provide an external interface to those cryptographic processes. Also, a most desirable implementation would be to make the trusted device tamperproof, to protect secrets by making them inaccessible to other platform functions and provide an environment that is substantially immune to unauthorised modification. Since tamper-proofing is impossible, the best
30 approximation is a trusted device that is tamper-resistant, or tamper-detecting. The trusted device, therefore, preferably consists of one physical component that is tamper-resistant.

Techniques relevant to tamper-resistance are well known to those skilled in the art of security. These techniques include methods for resisting tampering, methods for detecting

tampering, and methods for eliminating data when tampering is detected. It will be appreciated that, although tamper-proofing is a most desirable feature of the present invention, it does not enter into the normal operation of the invention and, as such, is beyond the scope of the present invention and will not be described in any detail herein.

5 The trusted device is preferably a physical one because it must be difficult to forge. It is most preferably tamper-resistant because it must be hard to counterfeit. It typically has an engine capable of using cryptographic processes because it is required to prove identity, both locally and at a distance, and it contains at least one method of measuring some integrity metric of the platform with which it is associated.

10 Figure 1 illustrates the motherboard 10 of an exemplary computer platform (not shown). The motherboard 10 includes (among other standard components) a main processor 11, main memory 12, a trusted device 14, a data bus 16 and respective standard control lines 17 and address lines 18, and BIOS memory 19 containing the BIOS program for the platform.

15 Typically, the BIOS program is located in a special reserved memory area, the upper 64K of the first megabyte of the system memory (addresses F000h to FFFFh), and the main processor is arranged to look at this memory location first, in accordance with an industry wide standard

20 The significant difference between the platform and a conventional platform is that, after reset, the main processor is initially controlled by the trusted device, which then hands control over to the platform-specific BIOS program, which in turn initialises all input/output devices as normal. After the BIOS program has executed, control is handed over as normal by the BIOS program to an operating system program, such as Windows NT (TM), which is typically loaded into main memory 12 from a hard disk drive (not shown).

25 Clearly, this change from the normal procedure requires a modification to the implementation of the industry standard, whereby the main processor 11 is directed to address the trusted device 14 to receive its first instructions. This change may be made simply by hard-coding a different address into the main processor 11. Alternatively, the trusted device 14 may be assigned the standard BIOS program address, in which case there is no need to modify the main processor configuration.

30 Although, the trusted device 14 is described as a single, discrete component, it is envisaged that the functions of the trusted device 14 may alternatively be split into multiple devices on the motherboard, or even integrated into one or more of the existing standard

devices of the platform. For example, it is feasible to integrate one or more of the functions of the trusted device into the main processor itself, provided that the functions and their communications cannot be subverted. This, however, would probably require separate leads on the processor for sole use by the trusted functions. Additionally or alternatively, although the

5 trusted device is described as a hardware device that is adapted for integration into the motherboard 10, it is anticipated that a trusted device may be implemented as a 'removable' device, such as a dongle, which could be attached to a platform when required. Whether the trusted device is integrated or removable is a matter of design choice.

The trusted device 14 comprises a number of blocks, as illustrated in Figure 2: a

10 controller 20 for controlling the overall operation of the trusted device 14, and interacting with the other functions on the trusted device 14 and with the other devices on the motherboard 10; a measurement function 21 for acquiring an integrity metric from the platform; a cryptographic function 22 for signing or encrypting specified data; and interface circuitry 23 having appropriate ports (24, 25 & 26) for connecting the trusted device 14 respectively to the data bus

15 16, control lines 17 and address lines 18 of the motherboard 10. Each of the blocks in the trusted device 14 has access (typically via the controller 20) to appropriate volatile memory areas 27 and/or non-volatile memory areas 28 of the trusted device 14.

For reasons of performance, the trusted device 14 may be implemented as an application specific integrated circuit (ASIC). However, for flexibility, the trusted device is

20 preferably an appropriately programmed micro-controller. Both ASICs and micro-controllers are well known in the art of microelectronics and will not be considered herein in any further detail.

One item of data stored in the non-volatile memory is a certificate 30, which is illustrated in Figure 3. The certificate 30 contains at least a public key 32 of the trusted device

25 14 and an authenticated value of a platform integrity metric 34 measured by a TP. Optionally, the trusted device 14 also contains an identity (ID) label 36 of the trusted device 14.

Where present, the ID label 36 is a conventional ID label, for example a serial number, that is unique within some context. The ID label 36 is generally used for indexing and labelling of data relevant to the trusted device 14, but is insufficient in itself to prove the identity of the

30 platform under trusted conditions.

The trusted device 14 is equipped with at least one method of reliably measuring some integrity metric of the computing platform with which it is associated. The integrity metric is acquired by the measurement function 21, which is illustrated in more detail in Figure 4.

- 10 -

The measurement function 21 has access to non-volatile memory 40 for storing a hash program 41, plus volatile memory 42 for storing a computed integrity metric 43, in the form of a digest. The hash program 41 contains instructions for computing the digest, in code that is native to the main processor 11. In addition, part of the measurement function 21 is configured to respond to the main processor 11 as if it were addressable memory, such as standard read-only memory, by sensing memory read signals addressed to the trusted device 14 and returning appropriate data. The result is that the main processor 11 sees the trusted device, for the purposes of integrity metric measurement, as a standard read-only memory.

In the preferred implementation, as well as the digest, the integrity metric includes a Boolean value 44, which is stored in volatile memory 45 by the measurement function 21, for reasons that will become apparent.

A preferred process for acquiring an integrity metric will now be described with reference to Figure 5.

In step 500, at switch-on, the measurement function 21 monitors the activity of the main processor 11 on the data, control and address lines (16, 17 & 18) to determine whether the trusted device 14 is the first memory accessed. Under conventional operation, a main processor would first be directed to the BIOS memory first in order to execute the BIOS program. However, in accordance with the present embodiment, the main processor 11 is directed to the trusted device 14, which acts as a memory. In step 505, if the trusted device 14 is the first memory accessed, in step 510, the measurement function 21 writes to volatile memory 45 a Boolean value 44, which indicates that the trusted device 14 was the first memory accessed. Otherwise, in step 515, the measurement function writes a Boolean value 44, which indicates that the trusted device 14 was not the first memory accessed.

In the event the trusted device 14 is not the first accessed, there is of course a chance that the trusted device 14 will not be accessed at all. This would be the case, for example, if the main processor 11 were manipulated to run the BIOS program first. Under these circumstances, the platform would operate, but would be unable to verify its integrity on demand, since the integrity metric would not be available. Further, if the trusted device 14 were accessed after the BIOS program had been accessed, the Boolean value 44 would clearly indicate lack of integrity of the platform.

In step 520, when (or if) accessed as a memory by the main processor 11, the main processor 11 reads the stored native hash instructions 41 from the measurement function 21 in step 525. The hash instructions 41 are passed for processing by the main processor 11 over the

data bus 16. In step 530, main processor 11 executes the hash instructions 41 and uses them, in step 535, to compute a digest of the BIOS memory 19, by reading the contents of the BIOS memory 19 and processing those contents according to the hash program. In step 540, the main processor 11 writes the computed digest 43 to the appropriate non-volatile memory location 42 in the trusted device 14. The measurement function 21, in step 545, then calls the BIOS program in the BIOS memory 19, and execution continues in a conventional manner.

Clearly, there are a number of different ways in which the integrity metric may be calculated, depending upon the scope of the trust required. The measurement of the BIOS program's integrity provides a fundamental check on the integrity of a platform's underlying processing environment. Other integrity checks could involve establishing that various other devices, components or apparatus attached to the platform are present and in correct working order. In one example, the BIOS programs associated with a SCSI controller could be verified to ensure communications with peripheral equipment could be trusted. In another example, the integrity of other devices, for example memory devices or co-processors, on the platform could be verified by enacting fixed challenge/response interactions to ensure consistent results. Also, although in the present embodiment the trusted device 14 utilises the data bus as its main means of communication with other parts of the platform, it would be feasible, although not so convenient, to provide alternative communications paths, such as hard-wired paths or optical paths. Further, although in the present embodiment the trusted device 14 instructs the main processor 11 to calculate the integrity metric, it is anticipated that, in other embodiments, the trusted device itself will be arranged to measure one or more integrity metrics.

Preferably, the BIOS boot process includes mechanisms to verify the integrity of the boot process itself. Such mechanisms are already known from, for example, Intel's draft "Wired for Management baseline specification v 2.0 - BOOT Integrity Service", and involve calculating digests of software or firmware before loading that software or firmware. Such a computed digest is compared with a value stored in a certificate provided by a trusted entity, whose public key is known to the BIOS. The software/firmware is then loaded only if the computed value matches the expected value from the certificate, and the certificate has been proven valid by use of the trusted entity's public key. Otherwise, an appropriate exception handling routine is invoked.

Optionally, after receiving the computed BIOS digest, the trusted device 14 may inspect the proper value of the BIOS digest in the certificate and not pass control to the BIOS if the computed digest does not match the proper value. Additionally, or alternatively, the trusted

- 12 -

device 14 may inspect the Boolean value 44 and not pass control back to the BIOS if the trusted device 14 was not the first memory accessed.

Figure 6 illustrates the flow of actions by a TP, the trusted device 14 incorporated into a platform, and a user (of a remote platform) who wants to verify the integrity of the trusted platform. It will be appreciated that substantially the same steps as are depicted in Figure 6 are involved when the user is a local user. In either case, the user would typically rely on some form of software application to enact the verification. It would be possible to run the software application on the remote platform or the trusted platform. However, there is a chance that, even on the remote platform, the software application could be subverted in some way. Therefore, it is anticipated that, for a high level of integrity, the software application would reside on a smart card of the user, who would insert the smart card into an appropriate reader for the purposes of verification.

At the first instance, a TP, which vouches for trusted platforms, will inspect the type of the platform to decide whether to vouch for it or not. This will be a matter of policy. If all is well, in step 600, the TP measures the value of integrity metric of the platform. Then, the TP generates a certificate, in step 605, for the platform. The certificate is generated by the TP by appending the trusted device's public key, and optionally its ID label, to the measured integrity metric, and signing the string with the TP's private key.

The trusted device 14 can subsequently prove its identity by using its private key to process some input data received from the user and produce output data, such that the input/output pair is statistically impossible to produce without knowledge of the private key. Hence, knowledge of the private key forms the basis of identity in this case. Clearly, it would be feasible to use symmetric encryption to form the basis of identity. However, the disadvantage of using symmetric encryption is that the user would need to share his secret with the trusted device. Further, as a result of the need to share the secret with the user, while symmetric encryption would in principle be sufficient to prove identity to the user, it would be insufficient to prove identity to a third party, who could not be entirely sure the verification originated from the trusted device or the user.

In step 610, the trusted device 14 is initialised by writing the certificate 30 into the appropriate non-volatile memory locations of the trusted device 14. This is done, preferably, by secure communication with the trusted device 14 after it is installed in the motherboard 10. The method of writing the certificate to the trusted device 14 is analogous to the method used to initialise smart cards by writing private keys thereto. The secure communications is supported

- 13 -

by a 'master key', known only to the TP, that is written to the trusted device (or smart card) during manufacture, and used to enable the writing of data to the trusted device 14; writing of data to the trusted device 14 without knowledge of the master key is not possible.

At some later point during operation of the platform, for example when it is switched
5 on or reset, in step 615, the trusted device 14 acquires and stores the integrity metric 43 of the platform.

When a user wishes to communicate with the platform, in step 620, he creates a nonce, such as a random number, and, in step 625, challenges the trusted device 14 (the operating system of the platform, or an appropriate software application, is arranged to recognise the
10 challenge and pass it to the trusted device 14, typically via a BIOS-type call, in an appropriate fashion). The nonce is used to protect the user from deception caused by replay of old but genuine signatures (called a 'replay attack') by untrustworthy platforms. The process of providing a nonce and verifying the response is an example of the well-known 'challenge/response' process.

15 In step 630, the trusted device 14 receives the challenge and creates a digest of the measured integrity metric and the nonce, and optionally its ID label. Then, in step 635, the trusted device 14 signs the digest, using its private key, and returns the signed digest, accompanied by the certificate 30, to the user.

In step 640, the user receives the challenge response and verifies the certificate using
20 the well known public key of the TP. The user then, in step 650, extracts the trusted device's 14 public key from the certificate and uses it to decrypt the signed digest from the challenge response. Then, in step 660, the user verifies the nonce inside the challenge response. Next, in step 670, the user compares the computed integrity metric, which it extracts from the challenge response, with the proper platform integrity metric, which it extracts from the
25 certificate. If any of the foregoing verification steps fails, in steps 645, 655, 665 or 675, the whole process ends in step 680 with no further communications taking place.

Assuming all is well, in steps 685 and 690, the user and the trusted platform use other protocols to set up secure communications for other data, where the data from the platform is preferably signed by the trusted device 14.

30 The techniques of signing, using certificates, and challenge/response, and using them to prove identity, are well known to those skilled in the art of security and will, thus, not be described in any more detail herein.

- 14 -

Referring now to Figure 7, a specific embodiment of the present invention will be described. In Figure 7, a host computer 100 has a main CPU 102, a hard disk drive 104, a PCI network interface card 106 and DRAM memory 108 with conventional ("normal") communications paths 110 (such as ISA, EISA, PCI, USB) therebetween. The network interface card 106 also has an external communication path 112 with the world outside the host computer 100.

The network interface card 106 is logically divided into "red" and "black" data zones 114,116 with an interface 118 therebetween. In the red zone 114, data is usually plain text and is sensitive and vulnerable to undetectable alteration and undesired eavesdropping. In the black data zone 116, data is protected from undetected alteration and undesired eavesdropping (preferably encrypted by standard crypto mechanisms). The interface 118 ensures that red information does not leak into the black zone 116. The interface 118 preferably uses standard crypto methods and electronic isolation techniques to separate the red and black zones 114,116. The design and construction of such red and black zones 114,116 and the interface 118 is well known to those skilled in the art of security and electronics, particularly in the military field. The normal communication path 110 and external communication path 112 connect with the black zone 116 of the network interface card 106.

The host computer 100 also includes a trusted module 120 which is connected, not only to the normal communication paths 110, but also by mutually separate additional communication paths 122 (sub-referenced 122a,122b,122c) to the CPU 102, hard disk drive 104 and the red zone 114 of the network interface card 106. By way of example, the trusted module 120 does not have such a separate additional communication path 122 with the memory 108.

The trusted module 120 can communicate with the CPU 102, hard disk drive 104 and red zone 114 of the network interface card 106 via the additional communication paths 122a,b,c, respectively. It can also communicate with the CPU 102, hard disk drive 104, black zone 116 of the network interface card 106 and the memory 108 via the normal communication paths 110. The trusted module 120 can also act as a 100VG switching centre to route certain information between the CPU 102, hard disk drive 104 and the red zone 114 of the network interface card 106, via the trusted module 120 and the additional communication paths 122, under control of a policy stored in the trusted module. The trusted module 120 can also generate cryptographic keys and distribute those keys to the CPU 102, the hard disk drive 104, and the red zone 114 of the network interface card 106 via the additional communication paths 122a,b,c, respectively.

- 15 -

Figure 8 illustrates the physical architecture of the trusted module 120. A first switching engine 124 is connected separately to the additional communication paths 122a,b,c and also to an internal communication path 126 of the trusted module 120. This switching engine 124 is under control of a policy loaded into the trusted module 120. Other components of the trusted module 120 are:

- a computing engine 128 that manages the trusted module 120 and performs general purpose computing for the trusted module 120;
 - volatile memory 130 that stores temporary data;
 - non-volatile memory 132 that stores long term data;
 - 10 • cryptographic engines 134 that perform specialist crypto functions such as encryption and key generation;
 - a random number source 136 used primarily in crypto operations;
 - a second switching engine 138 that connects the trusted module 120 to the normal communication paths 110; and
 - 15 • tamper detection mechanisms 140,
- all connected to the internal communication path 126 of the trusted module 120.

The trusted module 120 is based on a trusted device or module 14 as described in more detail above with reference to Figures 1 to 6.

With regard to crypto key generation and distribution, the trusted module 120 generates
20 cryptographic keys, using the random number generator 136, a hash algorithm, and other algorithms, all of which are well known, *per se*, to those skilled in the art of security. The trusted module 120 distributes selected keys to the CPU 102, hard disk drive 104 and the red zone 114 of the network interface card 106 using the additional communication paths 122a,b,c, respectively, rather than the normal communications paths 110. Keys may be used for
25 communications between the internal modules 102,104,106,120 of the platform over the normal communication paths 110. Other temporary keys may be used (by the network interface card 106 or CPU 102) for bulk encryption or decryption of external data using the SSL protocol after the trusted module 120 has completed the SSL handshaking phase that uses long term identity secrets that must not be revealed outside the trusted module 120. Other temporary keys
30 may be used (by the hard disk drive 104 or CPU 102) for bulk encryption or decryption of data stored on the hard disk drive 104 after those temporary keys have been created or revealed inside the trusted module 120 using long term secrets that must not be revealed outside the trusted module 120.

The trusted module 120 enforces policy control over communications between modules by the selective distribution of encryption keys. The trusted module 120 enforces a policy ban on communications between given pairs of modules by refusing to issue keys that enable secure communications over the shared infrastructure 110 between those pairs of modules.

5 Figure 9 illustrates a process by which the trusted module 120 can perform a watchdog function and 'ping' the modules 102,104,106 connected to the additional communication paths 122. The trusted module generates a challenge 142 and sends it to the CPU 102, hard disk drive 104 and red zone 114 of the network interface card 106 using the additional communication paths 122a,b,c, respectively. Each of the CPU 102, hard disk drive 104 and network interface
10 card 106 responds with a response 144a,b,c, respectively, on the respective additional communication path 122a,b,c to say whether the respective module is active, and preferably that the module is acting properly. The trusted module 120 notes the responses 144a,b,c and uses them as metrics in its responses to integrity challenges that are described above with reference to Figures 1 to 6.

15 Figure 10 illustrates the process by which incoming external secure messages are processed when the trusted module 120 is the only module in the platform with cryptographic capabilities. An external message 146 is received by the black zone 116 of the network interface card 106 using the external communication path 112. The network interface card 106 sends a protocol data unit 148 (to be described in further detail later) containing some data and
20 a request for an authentication and integrity check to the trusted module 120 using the normal communication paths 110. The trusted module 120 performs the authentication and integrity checks using the long term keys inside the trusted module 120 that must not be revealed outside the trusted module 120, and sends a protocol data unit 150 containing an 'OK' indication to the red zone 114 of the network interface card 106 using the additional communication path 122c.
25 The network interface card 106 then sends a protocol data unit 152 containing some data and a request for decryption to the trusted module 120 using the normal communication paths 110. The trusted module 120 decrypts the data using either temporary or long term keys inside the trusted module 120, and sends a protocol data unit 154 containing the decrypted data to the CPU 102 using the additional communication path 122a. The CPU then takes appropriate
30 action.

Figure 11 illustrates the process by which the CPU 102 requests a policy decision from the trusted module 120. This could be used, for example, when the CPU 102 must determine whether policy allows certain data to be manipulated or an application to be executed. This is described in more detail in another patent application having the same date of filing as the

present application. The CPU 102 sends a protocol data unit 156 containing a request to the trusted module 120 using the normal communication paths 110. The trusted module 120 processes the request 156 according to the policy stored inside the trusted module 120. The trusted module 120 sends a protocol data unit 158 containing a reply to the CPU 102 using the additional communication path 122a, in order that the CPU 102 can be sure that authorisation came from the trusted module 120. If the action is authorised, the CPU 102 takes the necessary action. Otherwise, it abandons the process.

Figure 12 illustrates an example of the control of policy over protected communications between the modules 102, 104, 106. All of the communications in this example use the additional communication paths 122. The red zone 114 of the network interface card 106 sends a protocol data unit 160 that is destined for the hard disk drive 104 to the trusted module 120 on the additional data path 122c. In the case where the policy does not permit this, the trusted module 120 denies the request by sending a protocol data unit 162 containing a denial to the network interface card 106 on the additional data path 122c. Later, the CPU 102 requests sensitive data from the hard disk drive 104 by sending a protocol data unit 164 addressed to the hard disk drive, but sent on the additional data path 122a to the trusted module 120. The trusted module 120 checks that the policy allows this. In the case where it does, the trusted module 120 relays the protocol data unit 164 to the hard disk drive 104 on the additional data path 122b. The hard disk drive 104 provides the data and sends it in a protocol data unit 166 on the additional data path 122b back to the trusted module 120 addressed to the CPU 102. The trusted module 120 checks that the policy allows this, and, in the case where it does, relays the protocol data unit 166 to the CPU 102 on the additional data path 122a.

Figure 13 illustrates the format of the data protocol units 178 by which data is passed over the additional communication paths 122. The data protocol unit 178 has:-

- an identifier field 168 indicating the type of the protocol data unit;
- a length field 170 indicating the length of the protocol data unit;
- a source field 172 indicating the source of the protocol data unit;
- a destination field 174 indicating the destination of the protocol data unit;
- and so on, including in many cases a data field 176.

Not all fields are always necessary. For example, assuming the policy of the trusted module 120 forbids it to relay key protocol data units that that did not originate within the trusted module 120, the CPU 102, hard disk drive 104 and network interface card 106 can therefore assume that keys are always from the trusted module 120. Hence, source and destination fields are unnecessary in key protocol data units - such protocol data units are

- 18 -

implicitly authenticated. The design and construction and use, *per se*, of protocol data units is well known to those skilled in the art of communications.

It should be noted that the embodiment of the invention has been described above purely by way of example and that many modifications and developments may be made thereto
5 within the scope of the present invention.

CLAIMS

1. A computing apparatus comprising:
 - a trusted hardware module (120);
 - a plurality of further hardware modules (102,104,106);
 - a shared communication infrastructure (110) by which the modules can communicate
- 5 with each other; and
 - a first communication path (122a;122b;122c), distinct from the communication infrastructure, by which a first one (102;104;106) of the further modules can communicate directly with the trusted module but cannot communicate directly with any other of the further modules.
- 10 2. An apparatus as claimed in claim 1, wherein the trusted module and the first further module each include a respective computing engine which partakes in the direct communication *via* the first communication path.
3. An apparatus as claimed in claim 1 or 2, wherein:
 - the first further module (102) is operable to supply to the trusted module a request (156)
- 15 for operation on data; and
 - in response to such a request, the trusted module is operable to generate a response (158) and to supply the response to the first further module *via* the first communication path (122a) and not *via* the shared communication infrastructure.
4. An apparatus as claimed in claim 3, wherein the trusted module includes means (132)
- 20 for storing policy information regarding such operations which can and/or cannot be permitted, and is operable to generate the response with reference to the policy information.
5. An apparatus as claimed in any preceding claim, wherein the trusted module is operable to generate an encryption and/or decryption key and to supply that key to the first further module *via* the first communication path and not *via* the shared communication infrastructure.

- 20 -

6. An apparatus as claimed in claim 5, wherein the first further module is operable to use the key for encryption and/or decryption of data communicated *via* the shared communication infrastructure.
7. An apparatus as claimed in any preceding claim, wherein the trusted module is operable to generate a challenge (142) and to supply the challenge to the first further module *via* the first communication path or *via* the shared communication infrastructure using encryption set up using the first communication path.
8. An apparatus as claimed in claim 7, wherein:
in response to the challenge, the first further module is operable to generate a response (144a, 144b, 144c) and to supply the response to the trusted module *via* the first communication path or *via* the shared communication infrastructure using encryption set up using the first communication path; and
the trusted module is operable to use the response in generating an integrity metric of the apparatus.
9. An apparatus as claimed in any preceding claim, wherein:
the first further module (106) has a zone (114) for private data and a zone (116) for non-private data; and
the first further module is operable to supply and/or receive data from/for the private data zone *via* the first communication path (122c) and not *via* the shared communication infrastructure.
10. An apparatus as claimed in claim 9, wherein the first further module is operable to supply and/or receive data from/for the non-private data zone *via* the shared communication infrastructure.
11. An apparatus as claimed in claim 9 or 10, wherein the first further module has an interface (118) between the private and non-private data zones which is operable to inhibit the passing of data from the private data zone to the non-private data zone.

12. An apparatus as claimed in any preceding claim, wherein the first further module is a network interface module (106).

13. An apparatus as claimed in any preceding claim, and including a second communication path (122a;122b), distinct from the communication infrastructure and the first communication path (122c), by which a second one (102;104) of the further modules can communicate directly
5 with the trusted module but cannot communicate directly with any other of the further modules.

14. An apparatus as claimed in claim 13, wherein:

the first further module (102) is operable to supply to the trusted module a request (164) for a transfer of data between the first and second further modules; and

10 in response to such a request, the trusted module is operable to generate a response (164) and to supply the response to the first or second further module (104) *via* the first or second communication path (122b), as the case may be, and not *via* the shared communication infrastructure.

15. An apparatus as claimed in claim 14, wherein the trusted module includes means (132)
15 for storing policy information regarding such transfers which can and/or cannot be permitted, and is operable to generate the response with reference to the policy information.

16. An apparatus as claimed in claim 14 or 15, wherein:

in response to an appropriate such transfer response, the first or second further module is operable to supply the data to the trusted module *via* the first or second communication path,
20 as the case may be; and

in response to the receipt of such data, the trusted module is operable to relay the data to the second or first further module, as the case may be, *via* the second or first communication path, as the case may be.

17. An apparatus as claimed in any of claims 13 or 16, wherein the second further module
25 is a main processor unit (102) of the apparatus or a non-volatile data storage module (104).

- 22 -

18. An apparatus as claimed in any of claims 13 to 17, and including at least a third communication link (122b), distinct from the communication infrastructure and the other communication links (122a,122c), by which at least a third one (104) of the further modules can communicate directly with the trusted module but cannot communicate directly with any other
- 5 (102,106) of the further modules.

19. An apparatus as claimed in claim 18, wherein the second further module is a main processor unit (102) of the apparatus and the third further module is a non-volatile data storage module (104).

TITLE

Communications between Modules of a Computing Apparatus

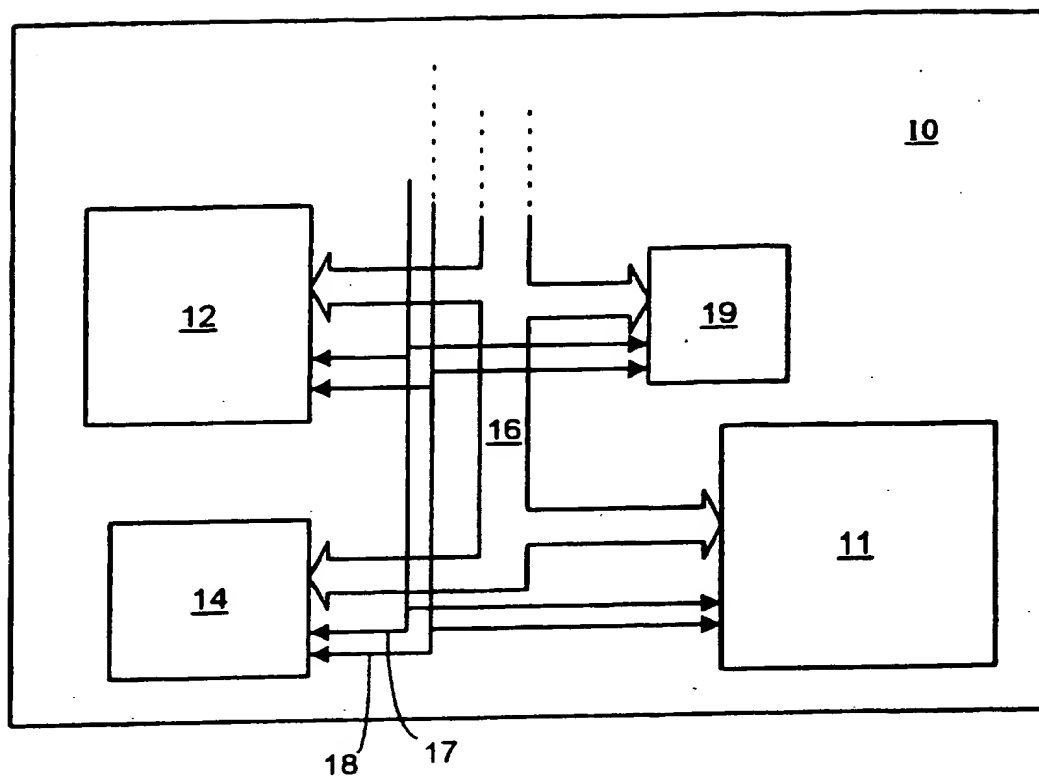
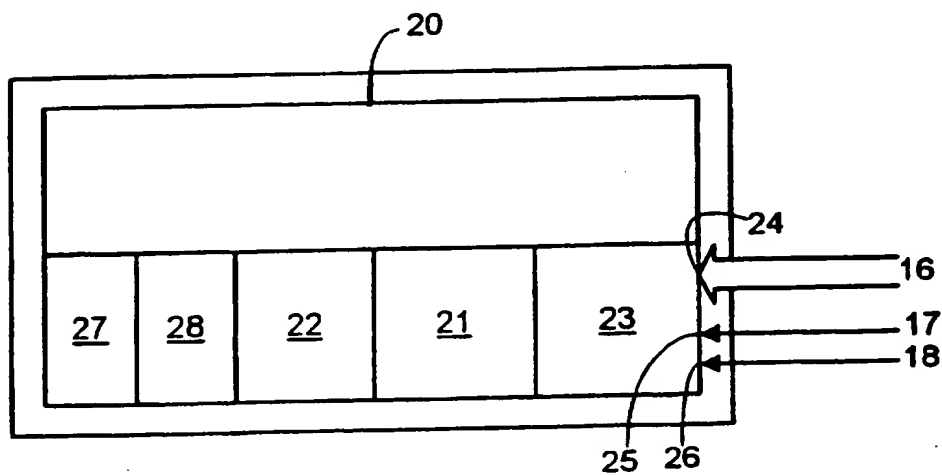
ABSTRACT

A computing apparatus comprises a plurality of hardware modules (102,104,106) and a shared communication infrastructure (110) by which the modules can communicate with each other in the usual way. In order to increase the level of trust and security in the apparatus, a trusted hardware module (120) is also provided and is connected to the other modules by
5 respective communication paths (122a;122b;122c), distinct from the communication infrastructure, by which each of those modules can communicate directly with the trusted module but cannot communicate directly with any other of the modules. The trusted module can therefore have secure communications, for example of "unsafe" data, with each of the other modules without any of the remaining modules eavesdropping, and the trusted module can route
10 unsafe data between any pair of the other modules, or decline to provide such routing, for example in dependence on policy stored in the trusted module.

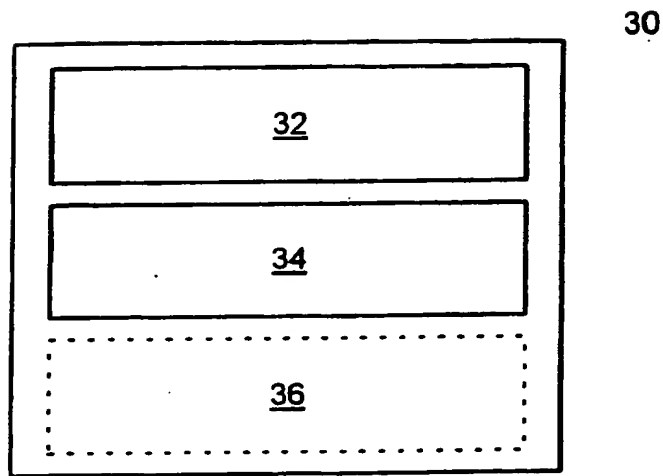
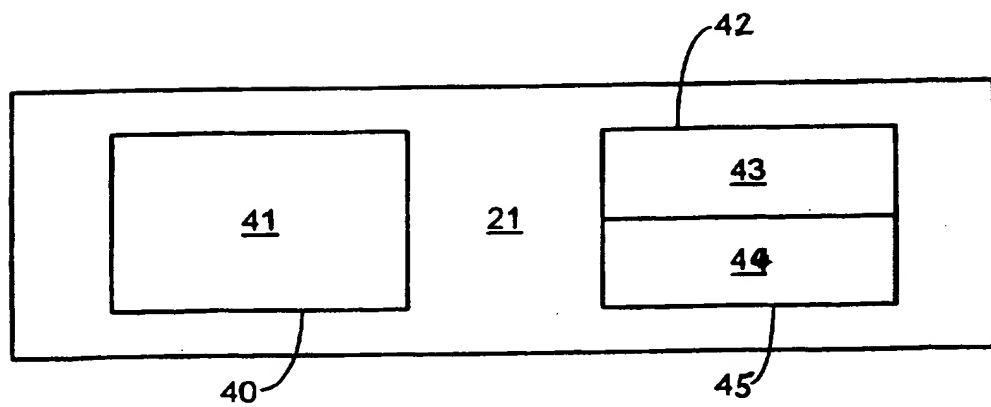
Figure 7.

THIS PAGE BLANK (USPTO)

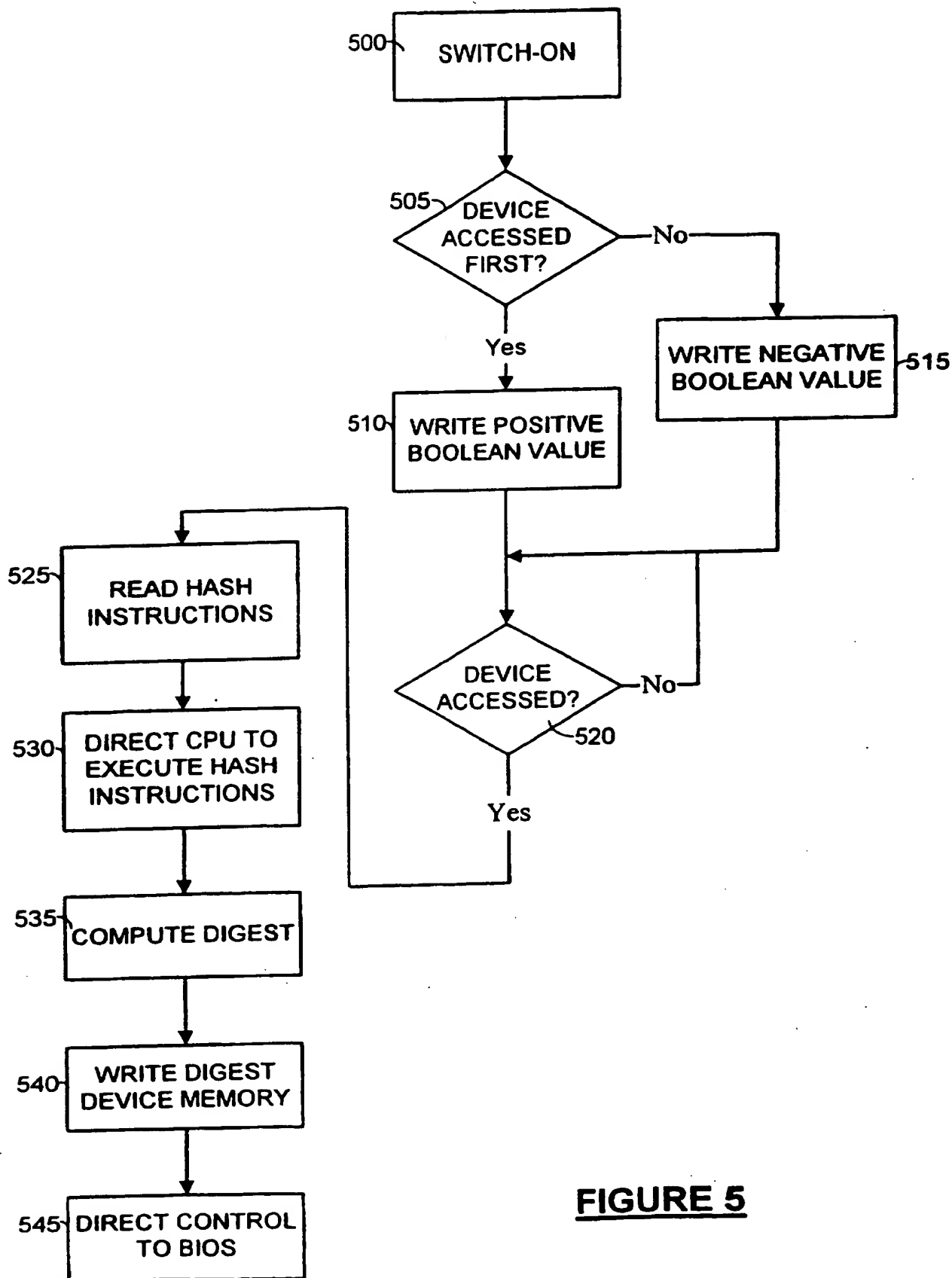
1/7

**FIGURE 1****FIGURE 2**

2/7

**FIGURE 3****FIGURE 4**

3/7

**FIGURE 5**

4/7

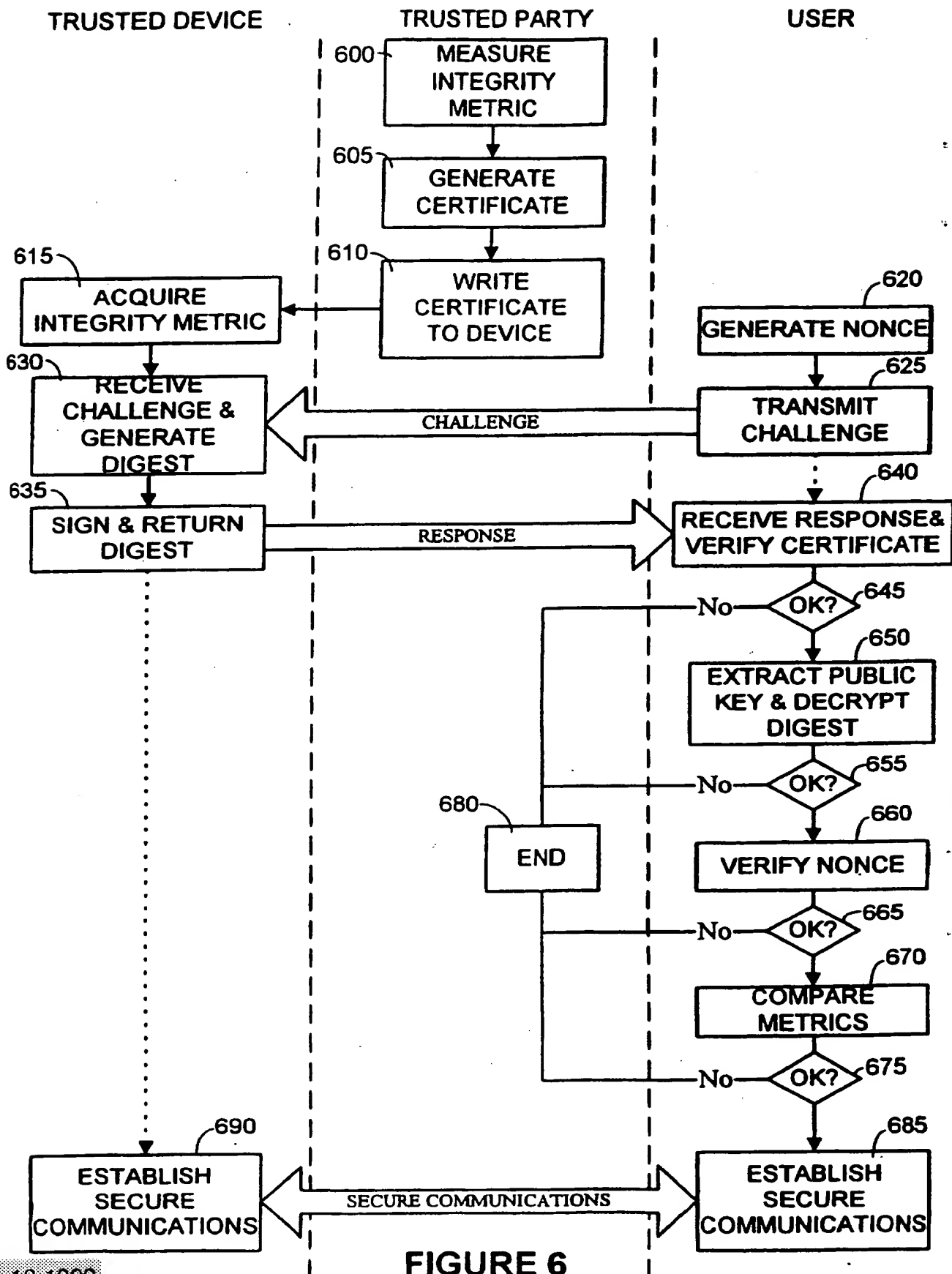


FIGURE 6

5/7

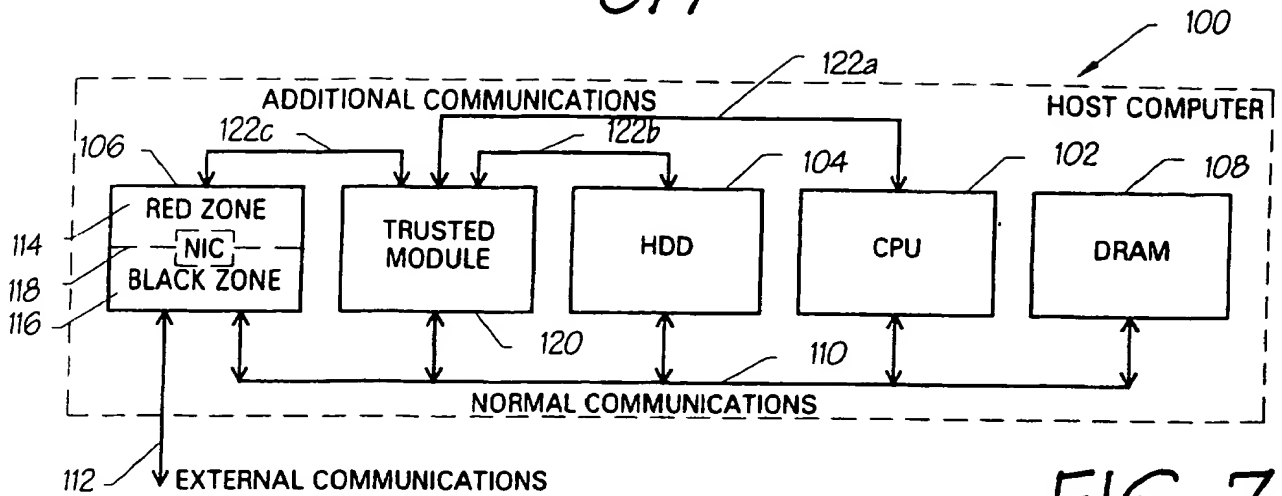


FIG. 7

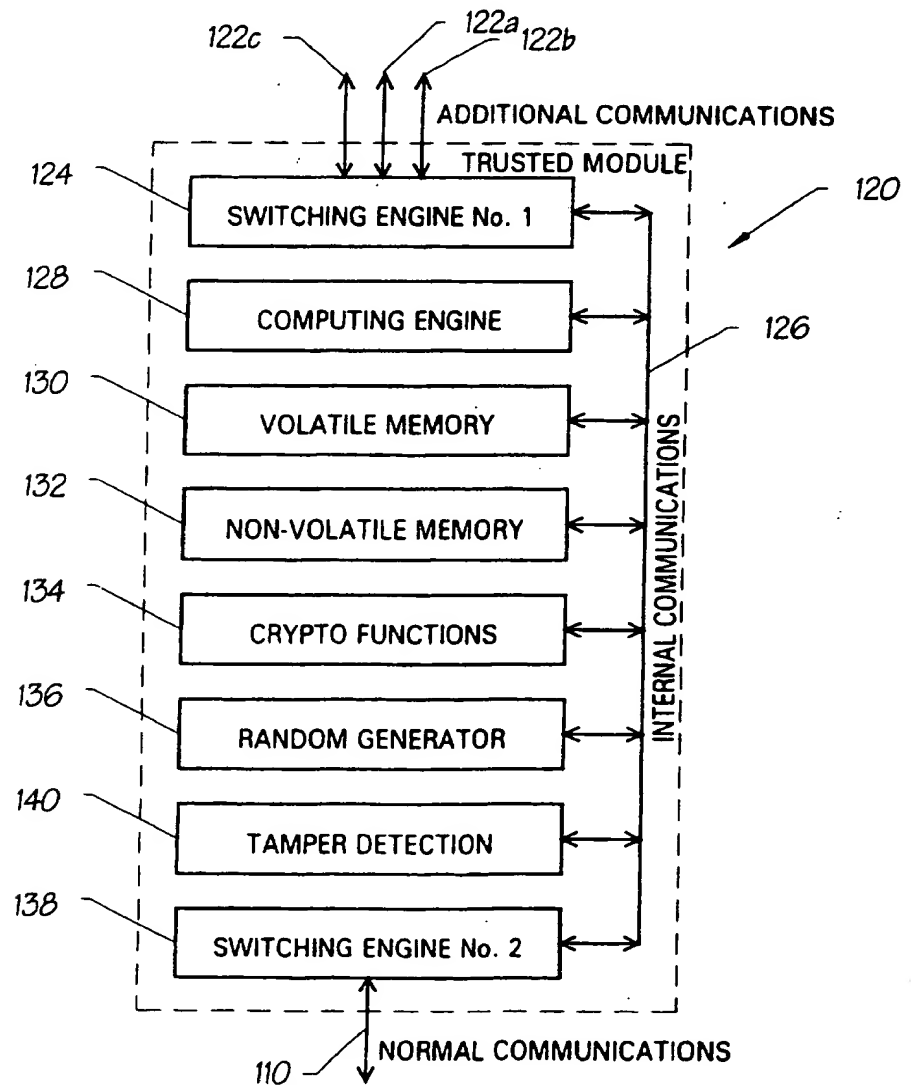


FIG. 8

6/7

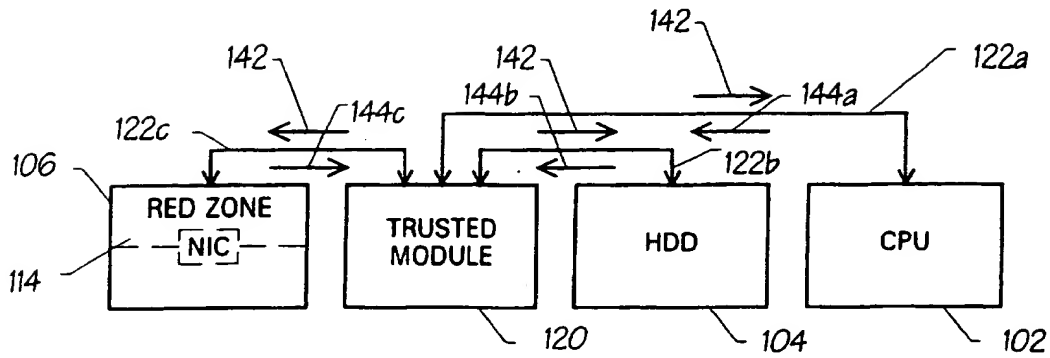


FIG. 9

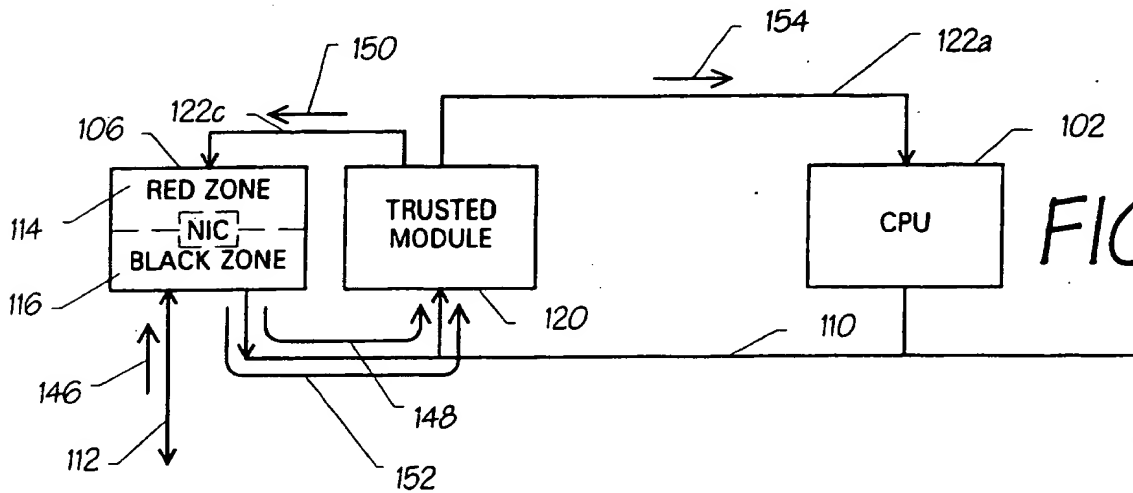


FIG. 10

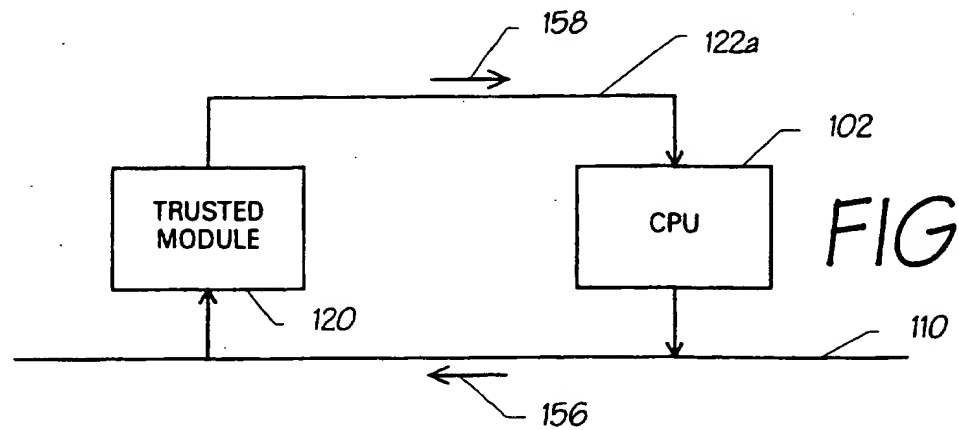


FIG. 11

7/7

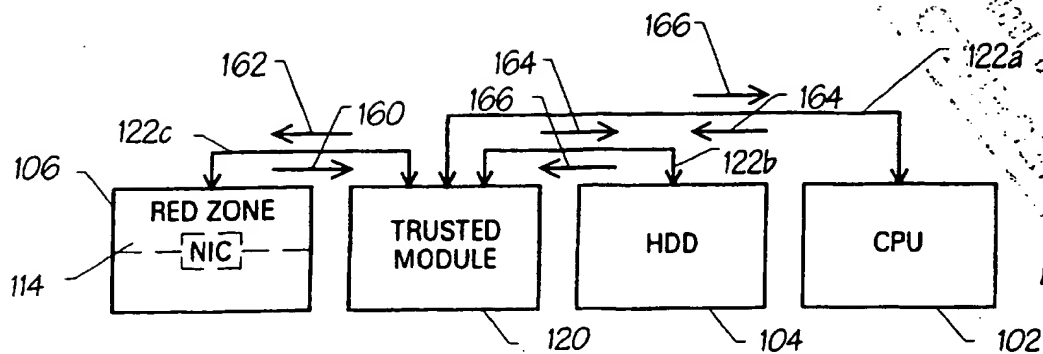


FIG. 12

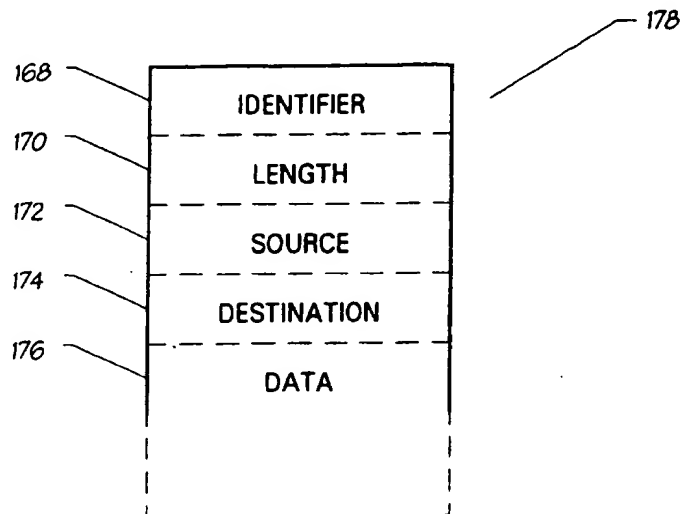


FIG. 13

THIS PAGE BLANK (USPTO)

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ **BLACK BORDERS**
- ☒ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☒ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER: _____**

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

THIS PAGE BLANK (USPTO)